Enhancing Static Charts with Data-driven Animations

Min Lu, Noa Fish, Shuaiqi Wang, Joel Lanir, Daniel Cohen-Or, and Hui Huang

Abstract—Static visual attributes such as color and shape are used with great success in visual charts designed to be displayed in static, hard-copy form. However, nowadays digital displays become ubiquitous in the visualization of any form of data, lifting the confines of static presentations. In this work, we propose incorporating data-driven animations to bring static charts to life, with the purpose of encoding and emphasizing certain attributes of the data. We lay out a design space for data-driven animated effects and experiment with three versatile effects, *marching ants*, *geometry deformation* and *gradual appearance*. For each, we provide practical details regarding their mode of operation and extent of interaction with existing visual encodings. We examine the impact and effectiveness of our enhancements through an empirical user study to assess preference as well as gauge the influence of animated effects on human perception in terms of speed and accuracy of visual understanding.

Index Terms—Visual encoding, Data-driven, Animated effects, Charts.

1 Introduction

Across the centuries, technological advancements have continuously spurred the creation of new means to present data to improve information sharing. Despite becoming increasingly more elaborate, lacking appropriate technology, visual charts had traditionally been designed to be displayed in hard copy form (*i.e.*, print). As personal computers, digital displays and smart phones have evolved and emerged onto the consumer market, hard copies made way for soft ones, changing the way we absorb and share information. Nowadays, visual charts are, more often than not, displayed on a digital screen, suggesting that traditional schemes can be augmented with animations to increase their effectiveness and impact.

When creating a chart to display pieces of information, one often wishes to communicate certain aspects of the data more than others. This is commonly achieved using color or geometric cues, whether intrinsic to the chart, such as by using magnitude, or extrinsic such as by using pointers. However, when the data is composed of multiple layers of information and different attributes need to be portrayed, employing the aforementioned cues to convey each and every attribute may lead to over-saturation and visual clutter.

Psychological and cognitive studies have shown that the human eye is particularly attuned to movement and motion, even more so than color transitions or changes in pattern or texture [1], [2]. This suggests that careful incorporation of dynamic elements within a static environment can be highly effective and has much potential for enhancing and highlighting selected data attributes.

In this paper, we explore data-driven animations, in which static charts are brought to life and enhanced with dynamic additions for the purpose of attribute emphasis or attention guidance. Given an existing visual chart depicting characteristically static data (*i.e.*, non-temporal data), we apply a suitable animation effect onto the displayed elements in order to highlight selected data attributes. Such enhancements help draw the attention of the viewer to important attributes, and are easy to design in a tasteful and minimal manner. They are particularly advantageous when commonly used cues have already been exhausted, or when one wishes to highlight aspects pertaining to motion (*e.g.*, directionality, traversal).

We present a general model for animated visual effects and exemplify it with three types of effects - *marching ants*, *geometry deformation* and *gradual appearance*, as these cater to a wide range of visual augmentations. Our focus is creating animated effects that augment existing static charts, and can be used to highlight or emphasize certain attributes of the data. For each of the visual effects, we provide details on their design dimensions and ways for data encoding. Our suggested animated visual effects are evaluated in a controlled user study in which static and animated versions of the same visual charts are shown to different participants, who are asked to use them in order to complete three tasks. Our findings indicate that animated effect additions act as constructive markers for data emphasis and contribute to faster understanding of the information conveyed in the charts, but are still subtle enough so as not to cause irritation.

The main contributions of the paper are as follows:

- A generic model for data-driven animations that enhance static visual charts
- Outlining three types of animated visual effects including a description of their design space, ways they can be used for data encoding and API code to support implementation
- A user study showing the potential benefit of adding animated effects to existing static charts.

2 A Data-driven Animation Example

This section shows an example that motivates the idea of data-driven animation exploiting our natural perception of movement [1] to deliver a message and help tell a story.

Figure 1 shows a visualization by William Playfair, hand-drawn in 1822, enhanced with one of our proposed data-driven effects,

- *Min Lu, Shuaiqi Wang, and Hui Huang are with Shenzhen University. E-mail: {lumin.vis, shuaiqiwang, hhzhiyan}@gmail.com.*
- *Joal Lanir is with The University of Haifa. E-mail: ylanir@is.haifa.il.*
- *Daniel Cohen-Or and Noa Fish are with Tel Aviv Univeristy. E-mail: {cohenor, noafish}@gmail.com.*
- *Hui Huang is the corresponding author of this paper.*

Marching Ants. The original, elegant yet static chart, visualizes the price of wheat (shown by bars) and weekly wages of labor (red line) over 250 years. Playfair intended his chart to convey the fact that "never at any former period was wheat so cheap, in proportion to mechanical labor, as it is at the present time". [1] However, from the static visualization, it is difficult to perceive the exact proportion of price of wheat to labor in every year. To help perceive the ratio of price to wage, the animated effect is placed over the space stretching between 'price of wheat' and 'wage', with a '$' sign serving as the moving element. The speed of the '$' sign is driven by the ratio between price of 'wage' and 'wheat', so that the faster it marches, the smaller the effort is. The moving elements help to see that during the 18th century the proportion of 'wheat' price to mechanical labor was indeed smaller, however, it also shows that this trend is becoming slower in the last three blocks. Please visit our project page to view the chart with the animation effects https://vizgroup.github.io/activateviz/.

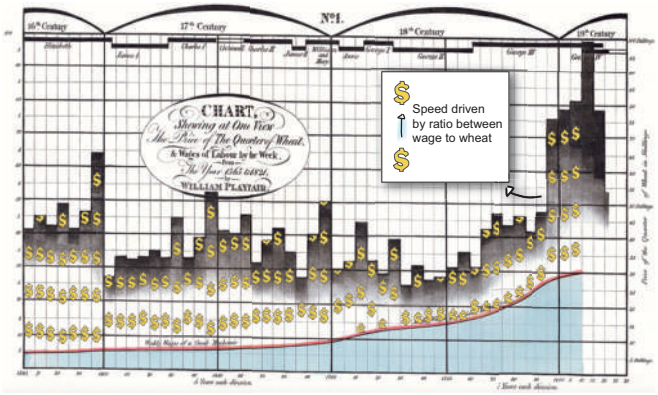

Fig. 1. Adding animated cues to Playfair's Visualization: moving '$' symbols are added over the bars. Their speed is mapped to the ratio of wage to wheat price. The faster moving '$' signs in 18th century strengthens Playfair's statement that "never at any former period was wheat so cheap, in proportion to mechanical labour, as it is at the present time".

3 Related Work

In this section, we first review the study of motion in visualizations. We then describe works that incorporated data-driven animations, that is, animations that are designed to encode attributes of the data, and provide a framework to classify these works. Finally, we describe systems and techniques that worked to add animation effects to graphics.

3.1 Animation and motion in visualizations

Motion perception has drawn substantial research interest in both the fields of cognitive psychology and visualization, and the strength of animated visual cues in attention attraction and memorization has long been established [1], [3]. Nakayama et al. [4] found that coherent motion facilitates group perception among individual elements, while Driver et al. [5] clarified that the oscillation in different coherent patterns excels at separating the visual elements into groups. These finding relate to the Common Fate Gestalt principle stating that elements are perceived as a group if they move together. Bartram et al. [6], [7], [8] characterized motion as a unique display dimension with substantial potential for its perceptual efficiency and interpretative richness. She performed several studies to better understand effective motion coding for information-rich interfaces, showing that motion coding can be used as a visual coding attribute, and that it does not interfere with existing color and form coding. Following Bartram's lead, Huber and Healey [9] studied the perceptual properties of motion: flicker, direction and velocity, optimizing motion configurations for efficient perception.

With the awareness of its potential perceptual benefits, animation has drawn much research attention and discussion in both the HCI and visualization fields. Fisher [10] discusses the opportunities and drawbacks of animation in visualization, and highlights that when the user is meant to compare the *before* to the *after*, animation is less likely to be of use. Tversky et al. [11] warn that animation is usually comprehended in a discrete manner and may be difficult to perceive accurately, so it may actually hinder different visual analytic tasks. Nevertheless, most researchers agree that animation and motion can be highly successful in attracting user's attention [12], and when used effectively, can be beneficial for various visualization tasks [13].

Chevalier et al. [13] derived five categorized roles of animation in data visualization based on an earlier categorization for animation in user interfaces by Baecker and Small [14]. According to their categorization, one role of animation is to keep the user in context, such as with the use of animated transitions to improve graphical perception of charts sharing the same data [15], [16], or when browsing through the visualization within the data level, e.g., transitioning between items in a radial node-link diagram [17]. A second role of animation is to support visual discourse, by supporting the narrative as well as by highlighting content. Traditionally, for highlighting of attributes or items in static charts, different visual glyphs, such as annotated labels [18] and statistical lines [19], [20] are used, requiring high-level cognition and perception. However, with the aid of animation, visual cues incorporating motion and movement can be used to attract and guide attention to various visualization attributes [2]. Several works have applied animated effects to attract attention replacing static cues [21]. For example, motion may help with visual search and notifications [6], as well as in detecting and identifying patterns [22]. In algorithm visualization, cuing (flashing) can be used to indicate that two elements have exchanged values [23].

3.2 Data-driven animation

Another role of animation as listed by Chevalier et al. [13], is data encoding. In data-driven animation, visual variables are manipulated over time to encode the underlying data attributes. Figure 2 presents a categorization of different methods employing data-driven animation. We distinguish methods according to two dimensions: *driving data* which describes whether the data attribute that is to be animated is temporal or non-temporal and *visual variable* which describes which visual variable is the data attribute mapped to.

The most natural encoding for animation is to show how data changes over time, i.e., temporal data. A considerable body of work has been carried out examining how to use animation to encode temporal data. A famous example for this is Gapminder (explored in [24]), which uses a bubble chart to show the change in countries' attributes over time. In the chart, bubbles represent

1. https://www.economist.com/christmas-specials/2013/10/07/worth-a-thousand-words

countries, attributes are mapped to the axes of the scatterplot, while the time dimension is shown using animation of the items. Similarly, TimeRider [25] also uses animation of items in a scatterplot, in this case, to show the change in patient's data over time. Besides position (as in Gapminder and TimeRider), other visual variables are also used to encode temporal information, as can be seen in Figure 2. Cao et al. [26] inflate both the shape and size of objects in its timeline according to the amount of incoming tweets. Similarly, the size of the visual object is often animated to encode volume over time, e.g., the explosion of Initial Coin Offering (ICO) [27]. Color is animated to show change in status of items, such as the average speed of vehicles in a current location on a map [28], Animated texture and orientation are often used in scientific visualizations such as in flow visualization. For example, a cyclical set of textures is used to visualize animating steady flow fields [29], [30], and in a temporal grid vector field, the orientation of glyph, such as wind barbs, is animated to encode the direction over time [31].

Looking at works animating non-temporal data, Romat et al. [32] examined and defined a model for animated edge textures (moving particles) of the node-link diagram, to encode the data attributes related to the graph's edges through animation. Similarly, Scheepens et al. [33] encode the direction of traffic flow by evenly moved particle system. Buschmann et al. [34] designed animated spiral texture on edges to show the direction. The similar idea is used by Blaas et al. [35] to show transition in a high-order graph.

As we can see, only few works examine how to use animation to encode non-temporal data. In the current work, we fill this gap and take a general look at data-driven animations of non-temporal data, providing a generic model for such visualizations, and describing three animated effects that can be used in various charts with different visual variables.

		Driving data	
		Temporal data	**Non-temporal data**
Visual variable	**Position**	Gapminder [24] TimeRider [25]	Animated edge effect [32] Traffic flow direction [33]
	Shape	Twitter volume [26]	
	Size	Twitter volume [26] ICO Volume [27]	
	Color	Travel status [28]	
	Texture	Flow field vis. [29, 30]	Edge direction [34] Transition in graph [35]
	Orientation	Grid field vis. [31]	

Fig. 2. Categorization of Data-driven Animations: examples of existing works or techniques are shown in the corresponding cells.

3.3 Systems for adding animation effects to graphics

The growing awareness for the importance of live illustrations and charts (*e.g.*, in scientific publications [36]) has attracted some research effort directed at adding animated elements to static graphics. Generally, we identify two types of approaches. The first conveys motion of objects with static sketches, such as the use of arrows in mechanical systems [37], a static continuous line to chain multiple state transitions in a graph [35], afterglow static effects in user interfaces to indicate transitions [38], or physics-inspired rigs that propagate the primary motion of elements to produce plausible secondary motion [39].

The second approach uses actual animation effects to top up the static image. Draco [40] is a sketch-based interface that allows users to add a set of animation effects to illustrations. As a follow up, Kazi et al. [41] propose seven motion amplifiers to craft animated illustrations containing exaggerated dynamics of stylized 2D animations. Borrowing the key concept from the traditional 2D brush model, the *motion brush* replaces the static brush image with a 3D animated scene comprised of geometry, information and motion [42].

These systems and techniques are primarily intended for illustrations and artistic purposes, and are generated based mostly on the designer's preferences and experience. They do not map data attributes to the motion effects and are therefore not intended and less suitable for data representation and information encoding. Our proposed animated enhancement of visual charts is data-driven and designed to organically represent and emphasize attributes of the data.

4 Data-driven Animations

In visual charts, data is mapped to elements that carry visual features. Common and widely used visual features include position, color, size, etc. As a complement to these encodings, we explore data-driven *animated visual effects* as a form of kinetic visual encoding geared toward information and attribute enhancement.

4.1 A generic model for data-driven animated effects

We describe a model for animated effects that enhance an existing static chart. Our model only describes animated enhancements that are data-driven. That is, it describes an enhancement to current existing data attributes that are already displayed in a static form. Thus, we do not refer to possible highlighting of specific items, labels or other non-data type additions, which could be useful but is out of the scope of this work. Another important point is that the animated effects that we suggest are inherently repetitive. Bartram has suggested that repetitive oscillating motions are advantageous since screen location can then be preserved [6]. Thus, in order for the chart to be able to retain its initial objective of being statically displayed (e.g., in a pdf format), and without adding user-initiated interactive elements, the animations we suggest are repetitive and only augment an existing chart.

Four essential design dimensions are identified in our model: *driving data* (what to encode), *visual variable* (what to animate), *visual proxy* (where to apply), and *dynamic function* (how to animate).

- **Driving data**: the data encoded in the animated effect. Although our model covers temporal-data driven animations, it focuses on advocating animations driven by non-temporal data. For example in Figure 1, the driving data is the ratio between price of 'wage' and 'wheat'.
- **Visual variable**: the visual variable to be animated. For example, *position* is the visual variable for the animation in Figure 1. Aiming to concentrate on the core idea of animated visual effects, in this work we consider the effect of animating only a single visual variable. This can be later used as a basis for compound effects. As enumerated in Figure 2, six main visual variables are summarized from Bertin's list [43], by merging the value and lightness as color for simplicity. The three animated effect examples in

this work provide a good coverage of the visual variable (to be introduced in Section 5).

- **Visual proxy**: the graphical content that is animated. For example, the '$' sign is the visual proxy of the animated effect in Figure 1. This content is not necessarily visible prior to enhancement, and may include implicit structural characteristics as long as they can be interpreted or modeled by a dynamic process, such as the node clique which is distorted in the node-link diagram in Figure 10(b). The design of the visual proxy can be quite flexible. One may wish to employ animated effects with a semantic meaning (e.g., '$' symbol to indicate a theme related to economics), or remain more abstract (e.g., using moving stripes).
- **Dynamic function**: describes the manner in which the visual proxy and the visual variable are mapped to the data and transform temporally. For example, in Figure 1, the dynamic function associates the speed of the '$' sign to the ratio between 'wage' and price of 'wheat' and positions its movement from the line to the top of the bar. The choice of a suitable dynamic function depends on the choice of visual variable. There basically exist two fundamental types of functions. The first imposes continuous changes with smooth transitions, while the second operates in a discrete, discontinuous manner, used for visual attributes that cannot be interpolated easily, such as textures, etc.

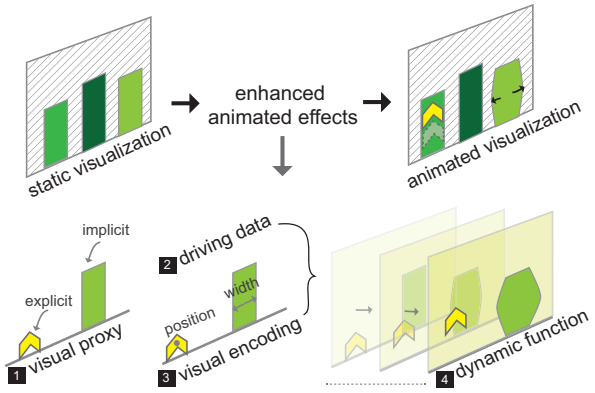

Fig. 3. Data-driven Animated Visual Effect: modelled as the visual encoding of a visual proxy, transforming its visual variable dynamically to represent certain data.

To better explain the model, consider an example in which a designer wishes to animate the size of a bar in a bar chart by flexing it back and forth in order to emphasize some attribute of the data. In this case, the driving data is the specific attribute of the data that is being emphasized, the visual proxy is the edges of the bar, the visual variable is its size, and the dynamic function associates the attribute to be emphasized with the amount of size to distort using the animation.

While the diverse choices of the four design dimensions create a large space of data-driven animations, the main categories of data-driven animated effects can be discussed according to the different combinations of *visual variable* and *driving data* as shown in Figure 2. Different kinds of animated effects can be generated with different choices of *visual variables* such as via changing the *position* of particles [32], [33], or using animated *textures* to show the transition in a graph [35] [34]. In this work, we exemplify the idea of non-temporal data-driven animations with three general effects with the use of different choices of visual variables, *Marching Ants* for position, *Geometry Deformation* for geometry-changing visual variables (i.e., shape, size), and *Gradual Appearance* for the appearance-changing visual variables (i.e., color, texture). Other types of animated effects using other visual variables may also exist, e.g., animating the orientation of a visual object (i.e., rotation) mapped to data.

5 Three specific types of animated Effects

We exemplify the idea of data-driven animations by closely examining three types of animated effects. The first, *marching ants*, shows a visual pattern repetitively moving along a certain path. This effect was chosen because it was used in previous works discussing animations, such as in flow and graph visualizations (e.g., [32], [33], [44]). We extend these works and show how it can be used in other types of charts (as shown in Figure 1) and discuss its overall design space. The second effect, *geometry deformation*, applies spatial modification to geometric properties of the visualization. This is similar to distortion-oriented techniques, but is applied according to data properties of the chart. The third effect, *gradual appearance*, applies discrete, repetitive changes to a visualization in order to highlight some order that appears within the visualization. This can be especially useful for highlighting hierarchies.

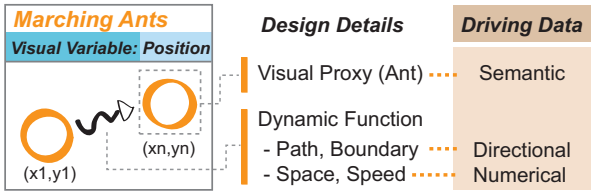

Fig. 4. Marching Ants: a *Marching Ants* effect is determined by five design components, which can be encoded to indicate numerical, directional and semantic messages.

5.1 Marching Ants

In *Marching ants*, a visual pattern (the visual proxy) shifts its position (the visual variable) along a path at a certain speed, creating an illusion of ants marching such that the movement of the position is driven by the data. Marching ants (*MA*) have been implemented in previous works of flow visualizations [33] and in graph visualizations encoding edge properties in a node-link diagram [32]). Here we generalize this group of animated effects and explore its possibility in encoding information more broadly. *MA* provides a perception of direction and speed, and, with appropriate design considerations, is also capable of conveying semantic meaning by way of an appropriately designed visual proxy (see Figure 5). *MA* is added as a layer on top of existing visual elements, and is orthogonal to them and mostly non-disruptive, and can thus be used to enhance an existing chart.

Marching ants instantiates the design model with five design components - *ant*, *path*, *boundary*, *marching speed* and *spacing* (Figure 4). *Ant* is the mobile visual proxy in the animated effect, and, while it originates from the well known traveling dashed line, it can take on any physical form and be used to convey semantic meanings. The remaining four design components define the manner in which the position of the ants changes (*i.e.*, dynamic function). *Path* defines the route taken by the *ant*. It provides an

intuitive perception of direction, but a notion of directionality is not a necessity for the application of *MA*. For instance, one can apply *MA* along a spiraling path inside a circle to create an animated texture effect without mapping to any directionality information. *Boundary* limits the marching ground of the *ant*, and along with *path*, delineates the area of operation of the *MA* effect. The *speed* of movement and *spacing* between ants creates an impression of pace and density, and can be mapped to encode numerical information pertaining to the underlying elements.

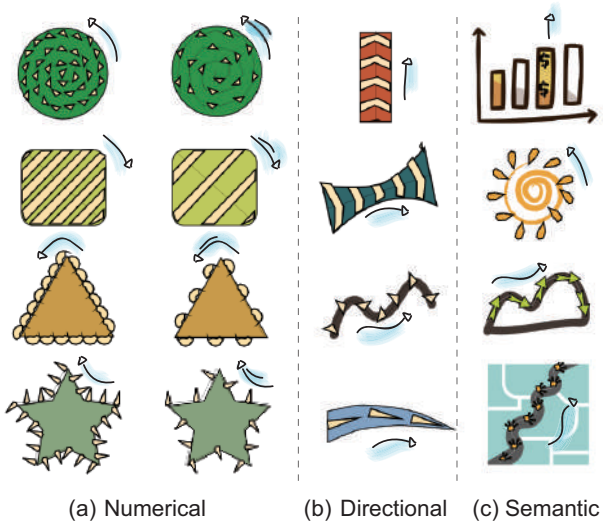

Fig. 5. Variation of Marching Ants: with various choices in design, *Marching Ants* demonstrates a rich expressive potential. Note that the arrows in hand-drawn style are markers to hint the animated effect due to the hard-print manuscript.

The five design components of *MA* create a rich space of variation that provides a multitude of creative options for this effect, thereby demonstrating its expressive power. As exemplified in Figure 5, *MA* can be flexibly layered over existing visual elements, and indicate numerical (a) and directional (b) information, as well as convey semantic meaning (c). By placing a path within an enclosed area (a), appealing animated effects can be created for element emphasis purposes, and by tuning the *speed* of the ants, and their *spacing*, one can better influence viewer perception and guide attention, while simultaneously representing numerical attributes. Naturally, *MA* is particularly suitable for cases where the data is inherently directional (e.g., in [33] showing the traffic direction). Finally, the design choices of the *ant* itself provide creative freedom that has value beyond fun and personal taste. As can be seen in (c), semantic properties can be visualized simply by a clever selection of graphics (*e.g.*, '$' sign at the top right).

Our model of *MA* extends the one described in Romat et al. [32] of animated edge textures in node-link diagrams. Speed and spacing in our model are similar to speed and frequency in Romat's model. However, through the *path* and *boundary* components our model enables to extend the path of particles beyond the restrictions of an existing edge to varied types of visualizations. Our model also enables different particle representations through the *ant* component.

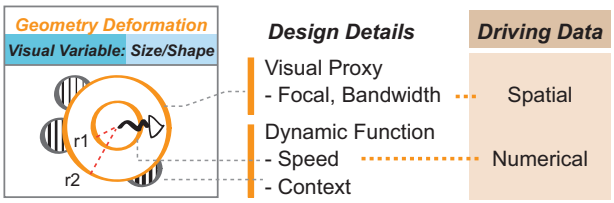

Fig. 6. Geometry Deformation: a *Geometry Deformation* effect is determined by four design components, which can encode the spatial structure and numerical information of the structure.

5.2 Geometry Deformation

Geometry Deformation (*GD*) is an animated effect that enhances the spatial structure of existing visual elements. It applies spatial modifications, which are potentially exaggerated, to the geometric properties of visual elements (*e.g.*, shape or size), at a certain pace. While related to distortion-oriented techniques (*e.g.*, focus+context [45]), rather than being used for user-initiated interactive actions that modify the visualization and its components (e.g., magnification in fisheye view [46]), *GD* is repetitively embedded within the visualization and is driven by the data, inducing varying degrees of deformation. This can implicitly convey the existence of groups and relationships and highlight their properties.

The perceptual principle that *GD* is based upon is the principle of common fate, which states that elements that are moving together tend to be perceived as a unified group. Bartram and Ware have shown that motions similar in path shape and phase are effective in perceptually grouping spatially scattered objects [6]. Thus, animation is useful to help in immediately recognizing associated elements [7]. They also suggested expansion/contraction as one of the repetitive motion types that can be easily discriminated [6]. *GD* uses these principles to highlight the spatial structure of the visualization (e.g., to highlight a clique or a group in a node-link diagram), encoding properties of a group (e.g., its size or tie-strength) to the speed of the animation.

In its model, *GD* is similar to fisheye lens [47], though intended for very different purposes. As Figure 6 shows, *GD* employs four design components. *Focal point* marks the center of the deformed visual proxy, and *bandwidth* defines the maximum reach of the proxy, from the *focal point*. *Context* - preserved or deformed, determines whether elements that do not take part in the effect itself remain unaffected (preserved) or are rather subjected to similar mechanisms as those that do (deformed). Finally, *speed* is the unique component in GD, compared with fisheye design, which controls the rate of deformation and can therefore convey numerical attributes of the underlying data elements.

Figure 7 summarizes the possibilities afforded by the four design components. Two original schemes (a,b) undergo *GD* with a small *bandwidth* (c,d), vs. a large *bandwidth* (e,f). *Context* preservation is exemplified in (e,f), vs. deformed in (g,h). Allowing context to be deformed lends an overall more organic look and feel to the visualization, and is suitable when the spatial relationship is implicit. For instance, *GD* applied on the matrix in (b) with *context* deformation generated the result in (h), where matrix cells neighboring the deformed element are affected by the deformation, promoting better spatial perception than that given by (f), where *context* was preserved. On the other hand, in cases where the spatial relations between elements are explicitly visualized, as in (a), sufficient visual cues exist to guide viewer perception, and

maintaining preservation of *context* results in a clearer scheme (e). The change in *bandwidth* and *speed* is driven by the numerical information of the spatial structure. For example, in (a), the size of the clique (yellow nodes) drives the deformation, such that a larger size induces a faster deformation with a larger bandwidth.

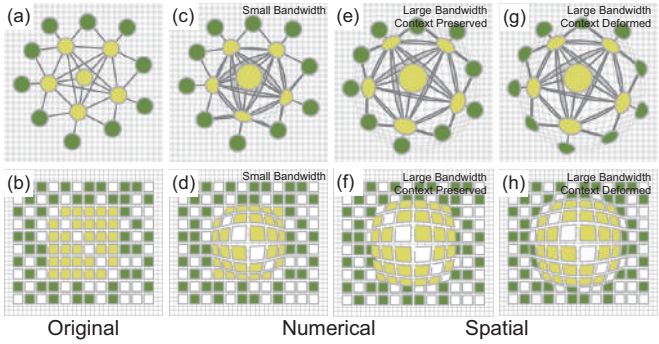

Fig. 7. Variation of Geometry Deformation: different design choices of *context preservation* and extent of *bandwidth* and *speed*, when enhancing the existence of cliques and their magnitude in a node-link diagram and matrix.

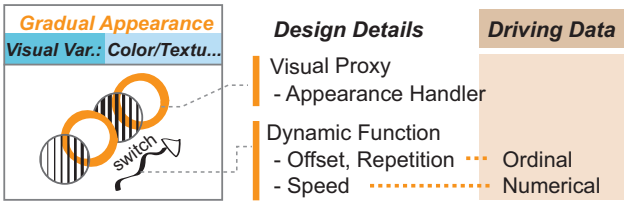

Fig. 8. Gradual Appearance: a *Gradual Appearance* effect is defined by four design components which are driven by the ordinal and numerical information.

5.3 Gradual Appearance

Gradual Appearance (*GA*) is a set of effects that apply repetitive changes to the visual variable of a proxy, such as brightness, saturation, color, etc. These effects deliberately operate at a low, non-continuous rate, so as to attract attention to the discontinuous change [48]. Activating appearance effects in set phases creates a sense of cohort among elements belonging to the same phase [9], and is thus useful for conveying hierarchy.

GA is reminiscent of cue techniques for object highlighting [45] (e.g., blinking or flickering). Cue techniques are anchored motions (do not change their position) which were found to be less distracting than other types of motions [49]. However, while these are typically utilized in interactive visual systems whenever the data satisfies a certain criteria, *e.g.*, searching, filtering, etc., in our case, *GA* effects are driven by the inherent attributes of the data, and are set up to encode this information visually. Moreover, applying a *GA* effect in phases facilitates the perception of element division into groups, where elements belonging to the same phase intuitively appear to form one group, sharing common attributes. For example, in a tree-based visualization, applying a *GA* effect over the nodes level by level, can be used to highlight the hierarchical structure of the tree.

As Figure 8 illustrates, a *GA* effect is defined by four design components: *appearance handler*, *offset*, *speed* and *repetitions*. *Appearance handler* is the visual proxy whose appearances are changed in the *GA* effect. It can be the full existing element whose transparency changes gradually by levels, or external borders flickered with different brightnesses, etc. *Offset* defines the lag between two consecutive effect applications. To effectively seize the attention of the viewer, the suggested offset between two applications should be greater than 500 ms [50], otherwise the rapid transition between two targets may compromise the perception of the second. *Speed* sets the appearance switching ratio of the effect, and is normally suggested to be no less than 120 ms per switch [9]. Lastly, *repetitions* determines the total loop number of effect applications.

The *GA* appearance handler supports the application of rich designs, for which any highlighting cue can be considered. Figure 9 presents a few representative designs. We identify two important considerations - space requirement and visual interference, and note their mutual contradiction. Space requirement is determined by the placement of the handler, such that a completely internal operation requires no additional space, and a completely external operation requires the largest amount of added space around the element. However, a visual handler layered over an element results in increased visual interference (or clutter) to existing features (*e.g.*, color, pattern, border), that is avoided when application is external. Naturally, one must consider the type of scheme at hand, for instance, applying external proxies to a treemap, which is an inherently compact scheme, is problematic due to space restrictions, but a circle packing diagram enjoys a sparse layout that is able to support extra features.

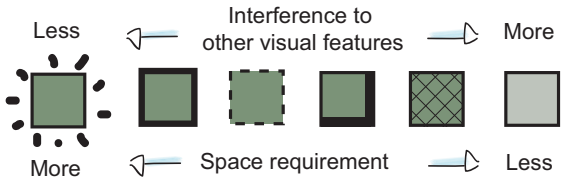

Fig. 9. Variation of Gradual Appearance: distribution of representative appearance handlers by considering the interference to other visual features and required space.

6 Implementation

The three visual effects and the examples shown in the next section are implemented in JavaScript, with external libraries including D3.js[2] and Paper.js[3]. We have designed a Javascript API to activate the animated effects by configuring its design components. For example, the following code fragment shows a simple call of the *Geometry Deformation* API, complying with its design model in Figure 6. The source code of the APIs and their examples are released on Github: https://github.com/vizgroup/DynamicEffect.

```
Visual_proxy = d3.select('path') //1. visual proxy
Focal = [50, 50] //2. focal: the center point of GD effect
Bandwidth = 50 //3. bandwidth: the extent GD amplifies
Speed = 0.9 //4. speed: the speed GD changes
geometryDeformation(Visual_proxy, Focal, Bandwidth, Speed)
```

Note that these APIs create animated effects on an individual visual proxy. The coordination of visual proxies in *GD* and *GA* (e.g., in treemap, rectangles of the same hierarchical level appear simultaneously) is maintained case by case. Here we show the

2. https://d3js.org/
3. http://paperjs.org/tutorials/

applications of *MA* and demonstrate how the data is mapped. Taking *Marching Ants* in a boxplot (Figure 10a) as an example, boxplot graphical items are traversed (Line 2-5) to instantiate design components (e.g., the boundary point list) which are then fed to the API '*marchingAnt*' as explained in Figure 4.

```
1  liMA= []
2  d3.selectAll('.boxplot')
3    .each(function(d, i){
4      //go over each boxplot to instantiate design components
5  })
6  MA_Start()
7  for(var ma in liMA){ //activate MA on each boxplot
8      marchingAnt(ma['ant'], ma['path'],
9      ma['boundary'], ma['speed'], ma['space'], ma['color'])
10 }
11 MA_End()
```

We utilize the power of Paper.js in processing vectors. The cartoon-style symbols in our work (e.g., '$' of Playfair's example in Figure 1) are designed in Adobe Illustrator, then loaded and manipulated via programming with the other *MA* API '*marchingAntByExample*', as shown below.

```
1  paper.project.importSVG('../svg/money.svg', function(item){
2      MA_Start()
3      for(var ma in liMA)
4          marchingAntByExample(visual_proxy, ma['path'], ma['boundary'],
5              ma['speed'], ma['space'], 'orange')
6      MA_End()
7  })
```

Note that except for the data driving *Marching Ants* in Figure 1 (i.e., the ratio between 'wage' and price of 'wheat'), which is manually recovered from Playfair's hand-made visualization, other driven-data is derived from the original data. Please refer to the Github repository for the complete code of these examples.

7 Examples

Conventional charts are tried-and-true tools that make good use of static visual cues to display information. Here, we demonstrate ways to extend static charts using animated effects, to promote better understanding or provide more information. All examples can be viewed on the project page: https://vizgroup.github.io/activateviz/. Figure 10 contains six examples of basic static charts that were enhanced with our three animated effects. In (a), two types of common charts are enhanced with *MA* to convey numerical information. Chart 1 is a boxplot with an added *MA* effect running up and down the top and bottom boxes of each plot in different speeds in order to convey the direction and compactness of the distribution. Chart 2 is a circos enhanced with *MA* over the top of its ribbons. The path of the ants convey the direction of the ribbon; the speed of the ants corresponds to the capacity of each ribbon, such that a larger capacity induces a faster movement. While the capacity is also visualized by the width of the ribbon, small differences between ribbons are easier to discern by comparing ant speed rather than ribbon width. In (b), we enhance two different graph visualizations with *GD* effects to emphasize the existence of cliques and their magnitudes. In both the node-link diagram and the matrix, the cliques are recovered from the underlying data and are considered as the focal regions for the operation of *GD*. The deformation bandwidth is driven by numerical attributes of the cliques, *e.g.*, clique size. This creates a visual connection between the perception of clique size and the illusion of pumping/contraction. In this case, the advantage of the animated effect over its static counterpart, is its ability to clearly emphasize the important attributes of the data by virtually making them pop out of the arguably messy layout of the diagram. Finally, in (c), two tree visualizations are enhanced with *GA* effects to convey hierarchical order. Given hierarchy-based data, nodes in the tree are wrapped with a blinking visual proxy whose blinking offset is driven by the level (depth) of the node in the tree. The *GA* effect creates an illusion of unity among nodes in the same level as they fade in and out, thereby providing viewers with an explicit in-order traversal of the data.

8 Evaluation

While various studies examined the perceptual properties of motion variables (e.g., [7], [32]), only few have empirically examined the benefits of animations over non-animated alternatives. Among these, the focus was on examining the animation of *temporal data* in various types of graphs, mostly comparing animation to small multiples [24], [51], [52]. Our work focuses on animating *non-temporal data*, and as such, to the best of our knowledge, this is the first empirical examination that investigates the benefit of adding animation to an existing chart.

Animated effects form a rich design space full of possibilities. This variation makes it difficult to estimate the general contribution of these effects as a group. Nevertheless, to assess the impact and effectiveness of animated effects, we performed a controlled user study designed to evaluate the contribution of the three types of animated effects mentioned above. We compared the performance of participants using six visualizations with and without animated enhancements. Our goal was to assess the potential of these enhancements to facilitate faster and more accurate visual understanding as well as elicit general personal preference appraisal.

Our main variable is therefore the *Animation* of the visualizations and has two values, either *Static* or *animated*. We examined user performance with six *Visualization types*, asking participants to complete three *Tasks* with each visualization type. Hence our study design is a 2 (Animation) x 6 (Visualization type) x 3 (Task) mixed design, with Animation being a between-subject variable and Visualization type and Task being within-subject variables.

8.1 Test visualizations

Six representative visualization schemes were chosen to appear in our experiment (see Figure 10) - two for each of our three experimental effects (*marching ants*, *geometry deformation*, *gradual appearance*). Boxplot and circos were chosen to be enhanced by *MA* since they comprise of numerical visualizations where the numerical reading baselines are shifted or curled for the benefit of the overall layout. For *GD*, we selected a node-link diagram and a matrix as they are two of the most common diagrams for network visualization, with explicit or implicit visual link structures. For *GA*, a treemap and a circle packing diagram were chosen for their popularity as visualizers of ordinal hierarchical information, showcasing either compact or less compact visual layouts.

8.2 Static vs. animated information enhancement

In order to evaluate our independent variable - extent of contribution of animated effects - we prepared two versions for each of the six visualizations, a reference and a controlled version. The reference version is completely static, while the controlled version augments it with animated effects as can be seen in Figure 10.

Several animated configurations were experimented with in order to decide on the speed of the animations. The update frequency of the animation in the boxplot was set ranging from

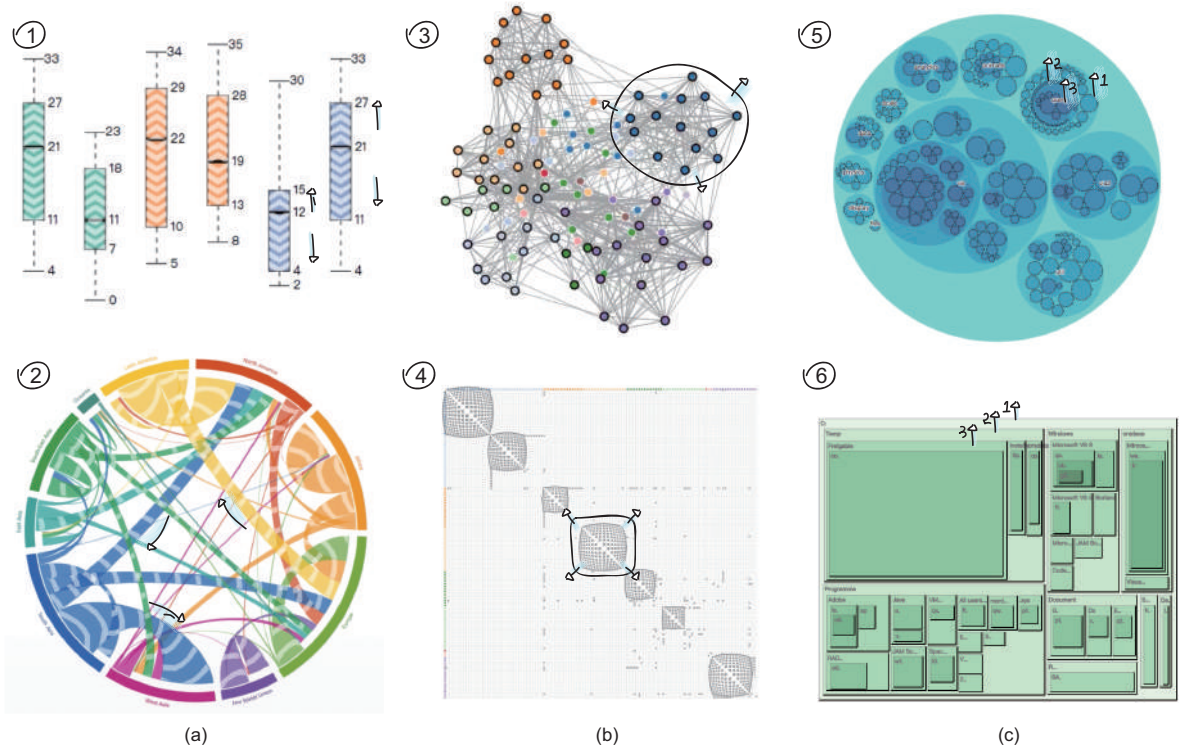

Fig. 10. Charts enhanced with animated effects: (a) *MA* driven by numerical attributes; boxplot (top), circos (bottom). (b) *GD* driven by the spatial structure of the data in the graph visualizations; node-link diagram (top), matrix (bottom) (c) *GA* driven by the hierarchy of the data in the tree visualizations; circle packing (top), treemap (bottom). Here the animated effects are represented by static cues in hand-drawn style (e.g., arrows). To view the original animated version, please visit the project homepage.

Visualization		Task Type	Specific Question
MA (Numerical)	Boxplot / Circos	Maximum/Minimum Identification	Which one has the largest ***X***? (***X*** = distribution compactness in Boxplot; transfer volume in Circos)
		Comparison	Which onehas larger ***X***, A or B?
		Sort	Sort groups by ***X***
GD (Spatial)	Node-link Diagram / Matrix	Maximum/Minimum Identification	Which clique contains the largest number of nodes?
		Comparison	Which clique contains more nodes, A or B?
		Sort	Sort cliques by the number of nodes
GA (Ordinal)	Circle Packing / Treemap	Maximum/Minimum Identification	Which node is in the deepest level?
		Comparison	Which subtree is deeper, A or B?
		Sort	Sort subtrees by depth

Fig. 11. Tasks and questions asked with each visualization type

1.2hz to 50hz (frequency was mapped to the data, so 1.2hz was mapped to the fastest instance) and 4hz to 50hz for circos. In GD, one deformation cycle is completed within a certain time. This was set for 1s for the node-link diagram and 4s for matrix. In GA, hierarchy levels are constantly added. In the case of both circle mapping and treemap we added a level every 0.5s.

8.3 Tasks

For each visualization, three types of questions were asked in order to measure participant performance in analysis tasks of different levels, all pertaining to the enhanced information. The first question asked the participant to identify the maximum/minimum element (numerical/structural/ordinal, respective to the three effect types). The second question asked for a comparison between two elements, while the third for a sorting of multiple elements. Table 11 presents the questions asked for each task and visualization type.

8.4 Measurements

For each task, both time cost and accuracy were measured to gauge the influence of static/animated enhancements on user perception in terms of effectiveness (time) and efficiency (accuracy) of the visual understanding.

8.5 Participants

We recruited 40 participants between the ages of 21 to 31 (M = 24.58, SD = 2.5), out of which 13 were Female and 27 were male. Participants were all students from a local university from various academic disiplines. Most participants were novices in data visualizations as reflected in their answer to a self reported question on data visualization experience (8 reporting no experience, 27 reporting little experience, 5 reporting a medium level of experience, and none reporting a high level of experience). All participants had normal or corrected-to-normal eye sight, and no participants were color blind (self-reported). Participants were paid a monetary compensation for their participation.

8.6 Procedure

Experiments were conducted in a quiet room, one participant at a time. Participants were seated in front of a 15" 4K InfinityEdge display. The participants were randomly divided into two groups, either *static* or *animated*. That is, each participant viewed and was tested on only one version of each visualizations - either the static or the animated.

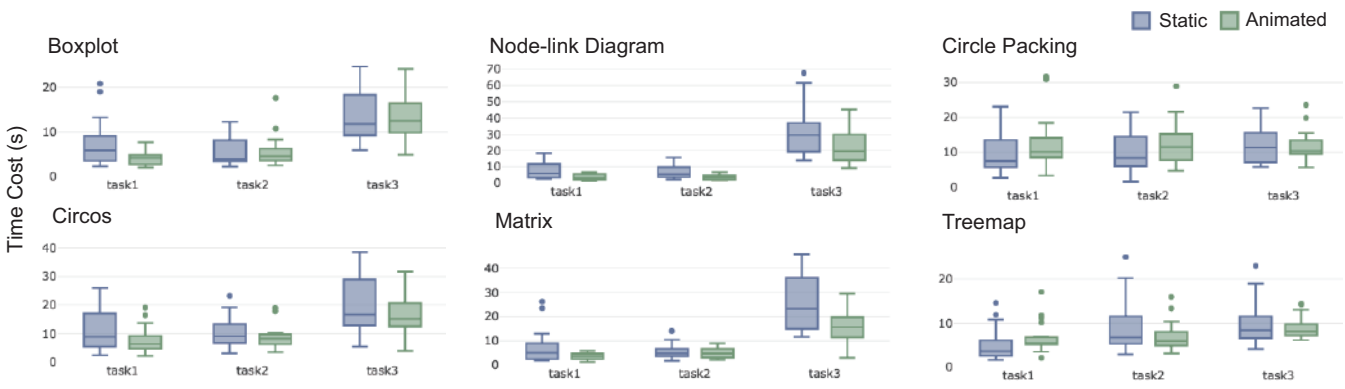

Fig. 12. Detailed time cost analysis of the animated vs. static effects in the six visualizations over three tasks. Animated effects were found to significantly improve the performance in Circos, Matrix, and Node-link diagram.

Each participant was accompanied by an evaluation assistant while performing the evaluation. The assistant gave a brief introduction to data visualization, and explained the outline of the experiment and its expected time frame. Participants were then presented with each of the six visualizations, one at a time. The order of the visualizations was randomized between the participants. Prior to seeing each of the visualizations, a short clarification was given regarding the relevant type of chart or diagram. The participant was presented with an example of the chart and was asked to answer an example question. The experimenter did not proceed until the participant verified that he or she completely understood the chart. Next, the participant was presented with a screenshot of the tested visualization, covered with a semi-transparency mask to prevent bias from early exposure, along with a motivating background story to the displayed information. Any questions raised were answered by the assistant, and the participant was free to proceed to the three task questions. For each one, the participant was first given a preview of the textual question first, so as to minimize reading comprehension issues. Having understood the question, the participant was free to proceed to the task itself, where the full-sized visualization was clearly presented in the center of the screen, along with the current question at the bottom. Elapsed time from participant first viewing the actual test until giving an answer was automatically recorded.

Finally, after completing all six visualizations, participants were presented with each of the visualization's counterparts, one at a time. That is, participants in the static condition where presented with the animated visualizations and vice-versa. For each visualization, participants were asked to say which one, animated or static, they prefer and why. The entire experimental session took about half an hour.

8.7 Results

A three-way ANOVA was conducted to investigate the impact of the animated effects on the different visualizations with Animation being a between-subject variable and Visualization type and Task being within-subject variables. Results indicate a significant main effect for Animation, $F(1,38) = 5.42$, $p=0.025$, $\eta^2 = .125$, with the animated condition being faster than the static condition. Additionally, there was a significant interaction effect between Animation and Visualization type, $F(5,34) = 5.48$, $p=0.001$, $\eta^2 = .447$.

To investigate this interaction further, we looked at each visualization type separately. Figure 12 displays the differences between the animated and static conditions according to the different visualization types and tasks. To investigate the difference between the visualizations, we ran six two-way mixed ANOVAs on each Visualization type with Animation being a between-subject variable and Task being a within-subject variable. Results indicate a significant main effect on Animation for Circos ($p=0.042$), Node-link diagram ($p=0.004$) and Matrix ($p=0.001$). It should be noted that for Boxplot, participants on average completed the task faster in the animated condition ($M=7.75$ sec) than with the static condition ($M=9.01$ sec). However, this difference was not significant.

8.7.1 *Task type*

To investigate the effect of Task type, we looked at each Task separately. We ran three two-way mixed ANOVAs on each Task type with Animation being a between-subject variable and Visualization type being a within-subject variable. Results indicate a marginal effect ($p=0.055$) for the first task (identify the minimum or maximum element), and a significant effect ($p=0.013$) for the third task (sorting multiple elements), indicating that for these two tasks, participants in the animated condition were faster than participants in the static condition.

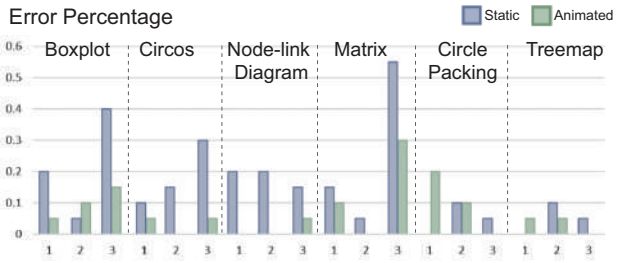

Fig. 13. Error rate of both the animated and static conditions over the six visualizations and three tasks

8.7.2 *Accuracy*

In terms of accuracy, there was an overall high accuracy rate, and in most cases, participants were able to name the correct answer in both versions. Figure 13 shows the error rate, which was calculated as the percentage of participants providing wrong answers in the

task (listed between 0 and 1). Summarizing over tasks, there were overall less errors in the animated condition (25 total errors) than in the static condition (53 errors). Outliers of lower accuracy were recorded in the sorting task in both versions of the Matrix visualization, for which it was more difficult to detect the size of all cliques in the Static condition (0.55 error rate, 11 errors out of 20) than the Animation (0.3 error rate). Other high-error tasks in the Static condition were the sorting tasks in both the Boxplot (0.4 error ratio, 8 out of 20) and the Circos (0.3 error ratio, i.e., 6 out of 20) visualizations which showed less errors in the animated condition.

8.7.3 Preference and subjective opinions

Preference information was asked in order to understand participant's attitudes toward the animated effects and help shed light on the reasoning behind participants' performance. Figure 14 summarizes the preference results of static (blue) vs. animated (green) visualizations. As we can see, there was a strong preference of participants toward the animated visualizations. This preference spanned across all visualization types, with node-link diagram being the strongest (of which 87 % of the participants preferred the animated condition).

Looking at the reasoning given by participants, their choices support our expectation of animated visual charts. The top three motives that were given were that animated effects clarify the information well, draw attention to important areas and are interesting (or fun). For information clarification, one participant commented that "animation is more suitable for integrating information in the visualization, especially when they are already visually cluttered". Regarding attention attraction, almost all participants agreed that "animation attracts attention well". Finally, some participants commented that "animation is more fun", and "animated visualizations are more impressive".

Regarding the fact that animated effects attract attention, some participants stated that this may be problematic if not implemented correctly, and distracted their attention from reading the charts in some of the charts. For example, one participant commented about the treemap chart that "the hierarchy of a treemap is better conveyed by nesting, the animation distracts my attention". Another participant commented on the boxplot that "it is hard to focus on the actual information because of the animation".

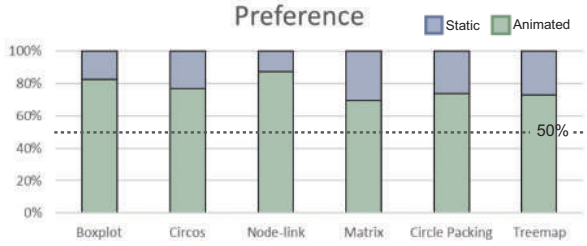

Fig. 14. Preference Voting for Static Visualization versus Animated Visualization (N=40)

9 Discussion

Our results suggest that animated effects can provide effective support for analysis tasks, taking less time compared to their static counterpart. Furthermore, there were overall fewer errors with the dynamic condition, and participants preferred the dynamic visualizations.

Both the *Marching Ants* and the *Geometry Deformation* effects showed improved performance for the animated condition over the static one. For *MA*, we observed during the experiment that some participants in the static group rotated their heads and pointed their fingers to the screen to better anchor the alignment of the to-be compared elements, while participants in the animated group rarely behaved in this way. This implies that *MA* potentially maintains more isotropic numerical encoding than static visual cues, such as length and width, which may facilitate more efficient and accurate numerical targeting and comparison, especially in the case of an unaligned or distorted layout, e.g., Circos. *GD* provides a more efficient enhancement to structure than static visual cuing when the visualization is cluttered, *e.g.*, in a node-link diagram. It acts as an orthogonal visual channel to encode numerical information along with structure, which promotes a more stable performance in spatially related analysis tasks.

Gradual Appearance did not show a measurable benefit over its static counterpart. A possible explanation for this is that in *GA*, each level is added over time, taking several rounds for the entire chart to be seen. This creates a disadvantage for the animated version over the static one which shows the entire chart with all hierarchy levels from the start. While participants preferred the animated version over the static one, this can be possibly affected by the novelty and "coolness" effect of animated visualizations. We still believe that the gradual highlighting of information in *GA* can be useful as a guide for chart reading by providing order identification, however, in the current study we were not able to show its benefits in respect to time or errors (error rates were low in both conditions). Future work should further examine whether this is so (possibly in more difficult tasks), and if this benefit can be measurable.

One potential area in which animated effects, especially *GA*, can be beneficial is story-telling [53] [54]. The advantage of animated effects in attracting attention to specific items and areas could potentially benefit the information narrative. With a proper design of the visual proxy, animated effects are able to help the visualization on self-contained information storytelling. For example, in Playfair's Wheat Chart (Figure 1), the visual proxy of *MA* is designed as 'money' symbols, which better supports the monetary narrative of the story. *Gradual Appearance*, in particular, can intrinsically lead readers' attention to data-driven sequential narratives that can be otherwise hidden in the data. For example, it is possible to apply *GA* onto a visualization's annotations in narrative order, in order to enhance the annotation's expressive power with animation [55]. Future work is needed to examine which kinds of animated effects are best effective, and how to best utilize them within storytelling.

9.1 Limitations and future work

In this work, we examined three specific data-driven animated enhancements. However, as we have shown, the design space of data-driven animated effects is large and there are many other possibilities. Future works should aim at developing more types of animated effects characterized by increased visual subtlety that does not compromise emphasis and attention guidance abilities. Taking it further, a more challenging problem may consider the combination of several animated visual effects that encode multiple information. It might be possible to animate two or more orthogonal visual variables simultaneously, e.g., position and size (e.g., circles changing their position and growing simultaneously). This can

generate compound animated effects. How to encode data to multiple visual variables and what kinds of simultaneous encoding are easier to discern remains future work.

Another avenue to explore in future work is user input and interaction. User input has not been considered in our current exploration. As an extension, user control can be combined to allow fine-tuning of the generated effects, both in data-binding aspects as well as in more cosmetic choices.

While animated effects can be highly effective, there are also several limitations and caveats that should be considered. Attention attraction and distraction are central to the idea of animation. In our work, the animated effects managed to effectively direct attention as planned, as we were able to show in the evaluation. However, much caution should be taken when using animation in visualizations [11]. Animated effects may easily cause attention distraction and disrupt the reading of other visual encodings when the enhancement via animated effects is not well controlled or when mapping of the effects are not aligned well with the underlying data. In more complex visualizations and when the visualization might include multiple encodings, the conjunction and separability of different visual encodings should be considered carefully. Animated effects can be perceptually burdening, and might narrow the focus in one direction to a single attribute. Using multiple animated effects in one visualization can also be distracting and create too much visual clutter. One way to alleviate these concerns is to use animated effects only at certain times, adapted according to the user task and context, rather than keeping them always active. Future studies can examine this direction.

Similar to other visual encodings, there are limitations of animated encodings for representing quantitative information that are probably dependent on each specific effect. It is also not entirely clear how accurate do we perceive these effects. These most likely are dependent on our perceptual abilities of viewing animations. In the experiment, we chose what we believe to be an acceptable configuration of parameters (such as speed) for the different examples. However, the perceivable quality and range of animation speed for the different effects, as well as the level of deformation for the GD effect, remains a follow-up issue (we note that for MA, user perception of some of the motion parameters were examined in [32]). Also, in our current work, animated effects are utilized as an *enhancement* to a certain encoding and are used as a second encoding to a certain attribute. It is unclear whether animation can be used as the only way to encode data. An investigation of this is out of the scope of this work but is an important topic to be studied in future. Finally, animated effects are of course only possible when viewed on a digital screen and cannot be supported in print. An important problem is how to design the legend for animated effects in such a way that it will be clear to the user. In our user study, we explained the effects verbally to participants. How to visually explain the mapping of an effect, whether statically or using animation remains to be explored.

10 Conclusion

In this paper, we explore the idea of data-driven animated visual enhancements, where commonly static visual charts are augmented with various animated effects. Our experimental findings indicate that the addition of these effects can promote faster understanding of the latent information in the underlying data. Possible explanations are that these effects are natural attention guides, or that they are able to visualize information that is otherwise hard to convey with static means.

With digital displays becoming mainstream, the incorporation of animated effects in visualizations is very likely to continue and develop. Just like color, with the upgrade from black&white to color print, movement is another intrinsic property of the real world. Thus, we believe that the ability to encode animation and add animation enhancement in information visualization will become more and more intuitive and prevalent.

References

[1] J. J. Gibson, *The ecological approach to visual perception*. Psychology Press, 2013.

[2] J. M. Wolfe and T. S. Horowitz, "What attributes guide the deployment of visual attention and how do they do it?" *Nature reviews neuroscience*, vol. 5, no. 6, pp. 1–7, 2004.

[3] A. Treisman, "Features and objects in visual processing," *Scientific American*, vol. 255, no. 5, pp. 114B–125, 1986.

[4] K. Nakayama and G. H. Silverman, "Serial and parallel processing of visual feature conjunctions," *Nature*, vol. 320, no. 6059, pp. 264–265, 1986.

[5] J. Driver, P. McLeod, and Z. Dienes, "Motion coherence and conjunction search: Implications for guided search theory," *Perception & Psychophysics*, vol. 51, no. 1, pp. 79–85, 1992.

[6] L. Bartram and C. Ware, "Filtering and brushing with motion," *Information Visualization*, vol. 1, no. 1, pp. 66–79, 2002.

[7] L. R. Bartram, "Enhancing information visualization with motion," Ph.D. dissertation, Simon Fraser University, 2001.

[8] L. Bartram, "Perceptual and interpretative properties of motion for information visualization," in *Proceedings of the 1997 workshop on New paradigms in information visualization and manipulation*, 1997, pp. 3–7.

[9] D. E. Huber and C. G. Healey, "Visualizing data with motion," in *VIS 05. IEEE Visualization, 2005.*, 2005, pp. 527–534.

[10] D. Fisher, "Animation for visualization: opportunities and drawbacks," *Ch*, vol. 19, pp. 329–352, 2010.

[11] B. Tversky, J. B. Morrison, and M. Betrancourt, "Animation: can it facilitate?" *International journal of human-computer studies*, vol. 57, no. 4, pp. 247–262, 2002.

[12] C. Ware and R. Bobrow, "Motion to support rapid interactive queries on node–link diagrams," *ACM Transactions on Applied Perception (TAP)*, vol. 1, no. 1, pp. 3–18, 2004.

[13] F. Chevalier, N. H. Riche, C. Plaisant, A. Chalbi, and C. Hurter, "Animations 25 years later: New roles and opportunities," pp. 280–287, 2016.

[14] B. Laurel and S. J. Mountford, Eds., *The Art of Human-Computer Interface Design*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[15] J. Heer and G. Robertson, "Animated transitions in statistical data graphics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1240–1247, 2007.

[16] Y. Wang, D. Archambault, C. E. Scheidegger, and H. Qu, "A vector field design approach to animated transitions," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 9, pp. 2487–2500, 2017.

[17] K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst, "Animated exploration of graphs with radial layout," in *Proc. IEEE InfoVis 2001*, 2001, pp. 43–50.

[18] D. Ren, M. Brehmer, B. Lee, T. Hollerer, E. K. Choe *et al.*, "Chartaccent: Annotation for data-driven storytelling," in *2017 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE, 2017, pp. 230–239.

[19] G. Carenini, C. Conati, E. Hoque, B. Steichen, D. Toker, and J. Enns, "Highlighting interventions and user differences: informing adaptive information visualization support," in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 2014, pp. 1835–1844.

[20] N. Kong and M. Agrawala, "Graphical overlays: Using layered elements to aid chart reading," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2631–2638, 2012.

[21] I. d. l. Torre-Arenas and P. Cruz, "A taxonomy of motion applications in data visualization," in *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*, H. Winnemoeller and L. Bartram, Eds. Association for Computing Machinery, 2017, p. 7.

[22] C. Ware and R. Bobrow, "Motion coding for pattern detection," in *Proceedings of the 3rd symposium on Applied perception in graphics and visualization*. ACM, 2006, pp. 107–110.

[23] B. Reed, P. Rhodes, E. Kraemer, E. T. Davis, and K. Hailston, "The effect of comparison cueing and exchange motion on comprehension of program visualizations," in *Proceedings of the 2006 ACM symposium on Software visualization*. ACM, 2006, pp. 181–182.

[24] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko, "Effectiveness of animation in trend visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008.

[25] A. Rind, W. Aigner, S. Miksch, S. Wiltner, M. Pohl, F. Drexler, B. Neubauer, and N. Suchy, "Visually exploring multivariate trends in patient cohorts using animated scatter plots," in *International Conference on Ergonomics and Health Aspects of Work with Computers*. Springer, 2011, pp. 139–148.

[26] N. Cao, Y.-R. Lin, X. Sun, D. Lazer, S. Liu, and H. Qu, "Whisper: Tracing the spatiotemporal process of information diffusion in real time," *IEEE transactions on visualization and computer graphics*, vol. 18, no. 12, pp. 2649–2658, 2012.

[27] "Animation: Visualizing the ico explosion, https://www.visualcapitalist.com/video-ico-explosion-one-animated-timeline/, accessed 10 july 2020."

[28] Z. Wang, T. Ye, M. Lu, X. Yuan, H. Qu, J. Yuan, and Q. Wu, "Visual exploration of sparse traffic trajectory data," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 1813–1822, 2014.

[29] H. Yu, C. Wang, and K.-L. Ma, "Parallel hierarchical visualization of large time-varying 3d vector fields," in *SC'07: Proceedings of the 2007 ACM/IEEE Conference on Supercomputing*. IEEE, 2007, pp. 1–12.

[30] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen, "Over two decades of integration-based, geometric flow visualization," in *Computer Graphics Forum*, vol. 29, no. 6. Wiley Online Library, 2010, pp. 1807–1829.

[31] R. V. Klassen and S. J. Harrington, "Shadowed hedgehogs: A technique for visualizing 2d slices of 3d vector fields," in *Proceeding Visualization'91*. IEEE, 1991, pp. 148–153.

[32] H. Romat, C. Appert, B. Bach, N. Henry-Riche, and E. Pietriga, "Animated edge textures in node-link diagrams: a design space and initial evaluation," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 2018, p. 187.

[33] R. Scheepens, C. Hurter, H. Van De Wetering, and J. J. Van Wijk, "Visualization, selection, and analysis of traffic flows," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 379–388, Jan 2016.

[34] S. Buschmann, M. Trapp, and J. Döllner, "Real-time animated visualization of massive air-traffic trajectories," in *2014 International Conference on Cyberworlds*. IEEE, 2014, pp. 174–181.

[35] J. Blaas, C. Botha, E. Grundy, M. Jones, R. Laramee, and F. Post, "Smooth graphs for visual exploration of higher-order state transitions," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 969–976, 2009.

[36] T. Grossman, F. Chevalier, and R. H. Kazi, "Bringing research articles to life with animated figures," *interactions*, vol. 23, no. 4, pp. 52–57, 2016.

[37] J. Heiser and B. Tversky, "Arrows in comprehending and producing mechanical diagrams," *Cognitive science*, vol. 30, no. 3, pp. 581–592, 2006.

[38] P. Baudisch, D. Tan, M. Collomb, D. Robbins, K. Hinckley, M. Agrawala, S. Zhao, and G. Ramos, "Phosphor- explaining transitions in the user interface using afterglow effects," in *Proceedings of the 19th annual ACM symposium on User interface software and technology*. ACM, 2006, pp. 169–178.

[39] N. S. Willett, W. Li, J. Popovic, F. Berthouzoz, and A. Finkelstein, "Secondary motion for performed 2d animation," in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. ACM, 2017, pp. 97–108.

[40] R. H. Kazi, F. Chevalier, T. Grossman, S. Zhao, and G. Fitzmaurice, "Draco: bringing life to illustrations with kinetic textures," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2014, pp. 351–360.

[41] R. H. Kazi, T. Grossman, N. Umetani, and G. Fitzmaurice, "Skuid: Sketching dynamic illustrations using the principles of 2d animation," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2016, pp. 4599–4609.

[42] A. Milliez, G. Noris, I. Baran, S. Coros, M.-P. Cani, M. Nitti, A. Marra, M. Gross, and R. W. Sumner, "Hierarchical motion brushes for animation instancing," in *Proceedings of the workshop on non-photorealistic animation and rendering*. ACM, 2014, pp. 71–79.

[43] J. Bertin, *Graphische Semiologie: Diagramme, Netze, Karten*. Walter de Gruyter, 2010.

[44] J. J. Van Wijk, "Image based flow visualization," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 745–754, 2002.

[45] Y. K. Leung and M. D. Apperley, "A review and taxonomy of distortion-oriented presentation techniques," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 1, no. 2, pp. 126–160, 1994.

[46] M. Sarkar and M. H. Brown, "Graphical fisheye views," *Communications of the ACM*, vol. 37, no. 12, pp. 73–83, 1994.

[47] G. W. Furnas, "Generalized fisheye views," *Acm Sigchi Bulletin*, vol. 17, no. 4, pp. 16–23, 1986.

[48] D. M. Hoffman, V. I. Karasev, and M. S. Banks, "Temporal presentation protocols in stereoscopic displays: Flicker visibility, perceived motion, and perceived depth," *Journal of the Society for Information Display*, vol. 19, no. 3, pp. 271–297, 2011.

[49] L. Bartram, C. Ware, and T. Calvert, "Moticons:: detection, distraction and task," *International Journal of Human-Computer Studies*, vol. 58, no. 5, pp. 515–545, 2003.

[50] J. E. Raymond, K. L. Shapiro, and K. M. Arnell, "Temporary suppression of visual processing in an rsvp task: An attentional blink?" *Journal of experimental psychology: Human perception and performance*, vol. 18, no. 3, p. 849, 1992.

[51] Y. Albo, J. Lanir, P. Bak, and S. Rafaeli, "Static vs. dynamic time mapping in radial composite indicator visualization," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, 2016, pp. 264–271.

[52] D. Archambault, H. Purchase, and B. Pinaud, "Animation, small multiples, and the effect of mental map preservation in dynamic graphs," *IEEE transactions on visualization and computer graphics*, vol. 17, no. 4, pp. 539–552, 2010.

[53] R. Kosara and J. D. Mackinlay, "Storytelling: The next step for visualization," *Computer*, vol. 46, no. 5, pp. 44–50, 2013.

[54] B. Lee, N. H. Riche, P. Isenberg, and S. Carpendale, "More than telling a story: Transforming data into visually shared stories," *IEEE Computer Graphics and Applications*, vol. 35, no. 5, pp. 84–90, Sep. 2015.

[55] J. Hullman, N. Diakopoulos, and E. Adar, "Contextifier: automatic generation of annotated stock visualizations," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2013, pp. 2707–2716.

Min Lu is an assistant professor at Shenzhen University. She received the BSc degree in computer engineering from Beijing Normal University, China, in 2011, and received the PhD degree on computer science at of EECS, Peking University in 2017. Her major research interests include visualization methodology and visual analytics. More information can be found at https://deardeer.github.io/

Noa Fish completed her BSc in Computer Science at the Technion in 2007. At Tel Aviv University, she then earned her MSc in 2013, and PhD in 2019. Her research interests include shape analysis and synthesis, and applying deep learning paradigms to solve problems with diverse visual aspects.

Shuaiqi Wang is a master candidate at the Visual Computing Research Center, Shenzhen University. His research interests include Visualization and Visual Analytics.

Joel Lanir is a senior lecturer and faculty member at the information systems department at the University of Haifa, Israel, where he heads the Human-Computer Interaction lab. He received his PhD in computer science from the University of British Columbia in 2009. His research interests lie in the fields of Human-Computer Interaction, Ubiquitous Computing and Information Visualization.

Daniel Cohen-Or received the BSc degree in both mathematics and computer science, in 1985, the MSc degree in computer science from Ben-Gurion University, in 1986, and the PhD degree from the Department of Computer Science, State University of New York at Stony Brook, in 1991. He is a professor with the Department of Computer Science, Tel Aviv University. His research interests include computer graphics, visual computing and geometric modeling and including rendering and modeling techniques, shape analysis, shape creation and editing, 3D reconstruction, photo processing, compression and streaming techniques, visibility, point set representation, morphing, and volume graphics.

Hui Huang is a distinguished professor of Shenzhen University, where she directs the Visual Computing Research Center. She received her PhD in Applied Math from The University of British Columbia in 2008. Her research interests span on Computer Graphics, Computer Vision and Visualization. She is currently a Senior Member of IEEE/ACM/CSIG, a Distinguished Member of CCF, an Associate Editor-in-Chief of The Visual Computer and is on the editorial board of ACM Trans. on Graphics and Computers & Graphics.